13th International Conference on Applications of Statistics and Probability in Civil Engineering, ICASP13
Seoul, South Korea, May 26-30, 2019

The UQLAB project: steps toward a global uncertainty quantification community

Stefano Marelli
Senior Scientist, Dept. of Civil Engineering, ETH Zurich, Zurich, Switzerland

Damar Wicaksono
Postdoctoral Researcher, Dept. of Civil Engineering, ETH Zurich, Zurich, Switzerland

Bruno Sudret
Professor, Dept. of Civil Engineering, ETH Zurich, Zurich, Switzerland

ABSTRACT: Now that the UQLAB software has gained significant momentum since its release three years ago, a new development stage has started. In addition to continuing the development of new features and improvements over the existing ones, the UQLAB project is now moving toward the creation of a global applied UQ community, by capitalizing on the ample userbase of the software and modern online resources. In this paper, we present the general development philosophy of UQLAB and how it has characterized its userbase. Additionally, we discuss the roadmap of the upcoming community development and the long-term goals of the project from a broader perspective.

1. INTRODUCTION

Computational modeling in engineering has become so prevalent that it is a key element in the design of any modern product or structure. Advancements in computational modeling and supercomputing architectures have improved the predictive capabilities of numerical simulations to unprecedented accuracy. Similarly, uncertainty quantification (UQ) has started to gather momentum at all stages of the lifetime of an engineering system: from design to maintenance, to disposal. At the same time, algorithmic advancements and the more recent adoption of methods borrowed from the machine learning community have enabled the application of uncertainty quantification to a wider class of problems.

Despite the latter, the widespread adoption of advanced UQ tools outside the purely academic practice seems to be lagging noticeably behind. This is not necessarily surprising, as the most promising UQ techniques often rely on complex abstractions, mixing advanced mathematical constructions, probability theory and statistics. Common examples include the statistical inference and representation of complex joint probability models with copulas (Nelsen, 2006), surrogate modeling techniques such as Gaussian process modeling (Santner et al., 2003) or generalized polynomial chaos expansions (Xiu and Karniadakis, 2002), sensitivity analysis techniques such as Sobol' indices (Sobol', 1993), modern reliability analysis techniques such as AK-MCS (Echard et al., 2011), and many others. While per se the adoption of complex techniques is of course not detrimental, it can restrict their usage to a selected audience of researchers highly trained in statistics and numerical methods. Unfortunately, this often means the exclusion of many others, who are highly trained on different aspects of the engineering practice and research.

In this setting, software plays a major role in enabling practitioners to apply advanced techniques without requiring extensive re-training. Indeed, the current widespread use of complex computational models is to a good degree due to the availability

of powerful software tools that provide a clear detachment between user interface and computational algorithms. Well-known, commercially successful examples include finite-element software such as Abaqus (Smith, 2009) or multi-physics software such as Comsol Multiphysics® (COMSOL, Inc, 2018). The landscape of UQ software, however, is still dominated by an archipelago of powerful, dedicated libraries and algorithms that cater to a specific community or to a high-skill target audience. This means, with few exceptions, different programming languages for different functionalities, the requirement of advanced software knowledge to deploy the tools provided, and small userbases. Note that, to some degree, such a fragmentation is inevitable: until now academic funding schemes have hardly provided resources to invest in software design and development, if not implicitly at the purely algorithmic stage. The impact potential of the proper implementation and documentation of accessible software is too often underestimated, as opposed to the continued development of minor improvements over existing methodological tools.

It is with this fragmentation in mind that the UQLAB project started at ETH Zürich in 2013 (Marelli and Sudret, 2014). In this paper, we provide first an overview of the development history and current status of the UQLAB software, followed by a peek at the long-term perspective of such a unique project and at its potential to foster a global applied UQ community.

2. THE UQLAB PROJECT

Conventional scientific software development focuses on providing well validated, reliable and stable functionality to highly trained interested users. Users are expected to either already have, or at least have access to, expertise and/or training facilities that can bring them up to the skill level or knowledge required to appropriately use the software. In a way, the target user is (either implicitly or explicitly) a researcher with a similar background as those developing the software itself. Examples of powerful, general-purpose UQ software that follow this development model include: the Dakota project developed at Sandia National Labs in the US, a collection of C libraries that cover a broad spectrum of UQ tools (Adams et al., 2014); the OpenTURNS initiative, developed in France by a partnership of private and public companies, with a similarly wide scope, but developed in Python instead (Andrianov et al., 2007); the OpenCossan software developed by the Institute of Risk and Reliability in Liverpool, a comprehensive Matlab-based software with a focus on reliability analysis and imprecise probability (Patelli et al., 2014). All of them are highly comprehensive, but their accessibility is limited to the trained audience of those who already are familiar with the tools they intend to apply. In each case, the user is provided with an extensive arsenal of libraries that can be employed to solve even the most complex UQ problems with state-of-the-art algorithms. Even disregarding the technical barriers posed by the need of directly accessing object-oriented libraries (not every applied science community is heavily focused on coding), this kind of software philosophy is unsuitable for non-experienced users who are drawn for the first time toward the field of UQ.

Therefore, this classical development model does not contribute to a second important function of scientific software, namely educating users. Bringing the audience into the design process, while well-known in the development of commercial software, is still uncommon for research-oriented software. Within this context, the UQLAB project was started in early 2013 at the Chair of Risk, Safety and Uncertainty Quantification in ETH Zürich with the design and development of the initial contents of a novel, general-purpose UQ software. Remarkably, the project started with, and is defined by, its own motto: *Make uncertainty quantification available for anybody, in any field of applied science and engineering* (source: `www.uqlab.com`). This means that the target audience, namely applied scientists and engineers from different research/application fields, was part of the design process since the very beginning. An important consequence of this choice was the need of designing both the user interface and the documentation model with a non-expert user in mind, who would learn how to properly pose and solve a UQ problem, alongside with learning the ropes of the UQLAB software itself.

After a closed-beta initial phase that lasted almost two years, the first open-source release of UQLAB was made available in April 2017. As of November 2018, with a total of over 1600 registered users from 72 countries worldwide, and over 130 citations of the official reference conference paper (source: Google scholar citations), UQLAB has become a well-recognized world-level player in the general UQ software community. In terms of software and development model, UQLAB has reached the maturity stage. At the same time, its wide adoption in largely different fields (ranging from theoretical physics to medical engineering, from computational macroeconomics to disease propagation) is a testimonial of the need of a software designed around the perspective users, rather than just around the profiles of the developers. Its self-imposed target of disseminating UQ outside the bounds of traditionally UQ-savvy fields has been successfully initiated.

It is at this stage that the next phase of the UQLAB project is starting: building an applied-UQ community centered around UQLAB. To say it with a cliché, the most powerful component of a software lies in its users. Indeed, beyond using the same software, they share the common objective of solving real world UQ problems, or developing new methodologies to solve them. Moreover, in the case of prevalently-academic software, the users are often characterized by uncommon know-how and unusually high motivation to take an active role in its continued development.

3. ON THE SOFTWARE SIDE: UQLAB FEATURES

3.1. A domain-specific language in Matlab

As mentioned in Section 2, UQLAB has been characterized since its inception by the inclusion of its desired audience within the design process. In particular, the highly unspecific class of "applied science and engineering" is primarily populated by researchers inexperienced in the topic of UQ and in the related methods. Therefore, a primary goal of UQLAB is to provide all the necessary tools to pose UQ problems in a consistent way, regardless of the particular application under consideration. This goal is achieved by the very nature of UQLAB: at its core lies a domain-specific-language (DSL) written in Matlab, designed to enforce a UQ-centric semantic structure to state the desired problem. The particular semantic model, first proposed in Sudret (2007) and De Rocquigny et al. (2008), is illustrated in Figure 1. According to it, for every type of UQ analysis up to three different ingredients can be identified:

A. A computational model, which is application-dependent, used to compute some quantity of interest (QoI) to the analyst (stresses, displacements, concentrations, etc.). This model is, in a more abstract sense, a black-box map of the form $\mathbf{y} = \mathscr{M}(\mathbf{x})$ that, for every combination of its input parameters $\mathbf{x}$, produces a set of model responses $\mathbf{y}$.

B. A model of the uncertainty in the input parameters due to natural variability (aleatory) or lack of knowledge (epistemic). This is classically some form of probabilistic model (in the form, e.g., of a joint PDF $f_{\mathbf{X}}(\mathbf{x})$). It is this uncertainty that causes the stochasticity in the model response.

C. An uncertainty analysis that aims at combining the variability of the input parameters with the computational model, to evaluate some property of interest of the stochastic model response, e.g. its mean, variance, or tail properties.

The syntax of UQLAB is centered on this clear-cut distinction between the different elements of every UQ problem. A positive consequence of "enforcing" this type of structure is that it helps the user to abstract their problem and clearly identify the various ingredients of this analysis.

Furthermore, repeating the same analysis on different computational or probabilistic models does not require any change to the structure of the analysis, as the three blocks identified earlier are essentially black boxes to each other. It is this last property that enables the highly modular architecture of UQLAB, highlighted in Figure 2. For the user, such a modularity implies that building complex uncertainty quantification frameworks is streamlined. It

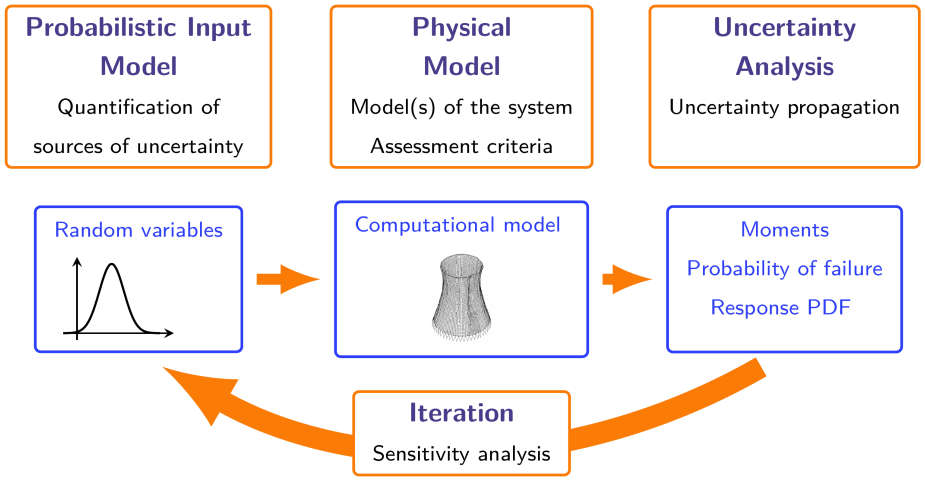

Figure 1: The general UQ framework for managing uncertainty (after Sudret (2007); De Rocquigny et al. (2008)) that defines the UQLAB *semantics*

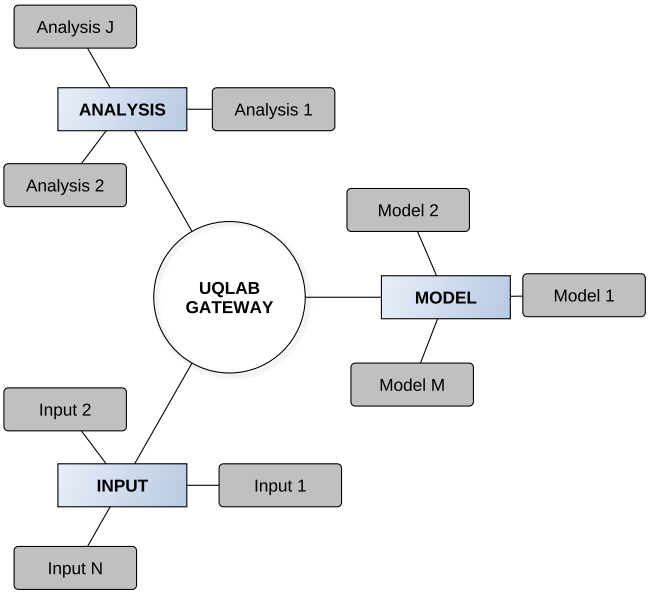

Figure 2: The modular structure of UQLAB*, modeled after Figure 1.*

also means that testing and comparing different methodologies to solve the same problem, a common occurrence during research, consists simply in substituting single blocks in Figure 2. As a concrete example, performing sensitivity analysis on a numerical model is as simple as defining a computational model, a probabilistic input model and a sensitivity analysis in three independent blocks of code. Subsequently, to test the performance of surrogate models in reducing the associated computational costs, it is sufficient to substitute the single computational model block with a surrogate one, without any further changes to the code describing the input variables and the sensitivity method.

3.2. A novel documentation model

Two ingredients are needed to enable a non-trained user to perform a proper UQ analysis: an intuitive and simple approach at posing the problem appropriately and an accessible documentation. While the first aspect was discussed in the previous section, the latter is a distinctive feature of UQLAB, and is achieved by a three-pronged approach:

- *Learn-by-example:* each type of input/model/analysis that is available within the software ships with an extensive set of examples that grow in complexity, covering the basics up to some of the advanced features, in easily relatable, formatted packages. The user is encouraged to browse through the examples, and modify them to suit their needs. In addition, extensive usage information is provided directly through the command prompt with the builtin Matlab documentation command `help`.

- *A unique set of user manuals:* more than a third of the UQLAB development time is devoted to the design and writing of its user manuals. Each manual is a self-contained document that comprises three complementary

parts. In the introductory part, a summary of the theory behind the methods is given, together with selected references to the relevant literature. In the second part, the same methods are showcased on simple but representative benchmark problems, following the structure of the theory part as closely as possible. The user is shown how to access each method and its variations, and how to access the important information produced by the analyses. Plenty of references to the relevant theory described in the first part are provided. The third part comprises a more standard software-reference list, where all the available options are provided in a user-friendly format.

- *The* UQLAB *website:* the final source of information for the end users is the UQLAB website (`www.uqlab.com`), which provides an easy-to-navigate hub containing information about the project itself, but also all the examples that ship with the software, with additional inter-links. It also offers accessibility to all of the documentation directly online.

Thanks to the combination of the intuitive language provided by UQLAB and its advanced documentation model, typical users can usually carry out their first UQ analysis within one to two hours.

3.3. UQLAB *features*

A last crucial point for a general-purpose UQ software is the breadth of its features. Due to its modular structure described in Section 3, UQLAB is an ideal environment to develop new content that seamlessly integrates with the rest of the available features. Indeed, adding additional surrogate models, or sensitivity measures, or even completely new modules that cover nonexistent functionality is straightforward.

At the time of writing, the current public version V1.2 comprises the following features:

- *Computational models:*
  - Support for m-files, strings and function handles, vectorized or not
  - UQLINK: an easy-to-deploy tool to connect to UQLAB external solver software that supports text-based configuration files (Abaqus, Comsol, OpenSees etc.)
  - A surrogate modeling tool that includes:
    - Polynomial chaos expansions (Xiu and Karniadakis, 2002; Blatman and Sudret, 2011)
    - Gaussian process modeling (Kriging) (Santner et al., 2003; Rasmussen and Williams, 2006)
    - Polynomial-Chaos-Kriging (Schöbi et al., 2015)
    - Low-rank tensor approximations (Konakli and Sudret, 2016)
    - Support vector machines for classification and regression (Vapnik, 1995; Schölkopf and Smola, 2002)
- *Probabilistic input models:*
  - Standard and custom marginal specification
  - Non-parametric marginals (kernel-smoothing)
  - Gaussian copula (Nelsen, 2006; Lebrun and Dutfoy, 2009)
- *Uncertainty analysis:*
  - Reliability analysis/rare-event estimation (Ditlevsen and Madsen, 1996; Lemaire, 2009)
    - FORM and SORM approximation
    - Simulation methods: Monte-Carlo simulation (MCS), importance sampling and subset simulation
    - Metamodel-based adaptive methods: active-learning Kriging MCS (AK-MCS, Echard et al. (2011)) and active-learning PC-Kriging MCS (APCK-MCS, Schöbi et al. (2017))
  - Sensitivity analysis (Saltelli et al., 2008; Le Gratiet et al., 2017)
    - Linearized methods: perturbation method, Cotter sensitivity factors (Cotter, 1979), standard regression coefficients
    - Correlation-based indices
    - Morris sensitivity factors
    - Borgonovo moment-independent indices (Borgonovo, 2007)

    - Sobol' (Sobol', 2001; Sudret, 2008), ANCOVA (Sudret and Caniou, 2013) and Kucherenko variance-decomposition based indices (Kucherenko et al., 2012)
- Bayesian inversion (Tarantola, 1987; Kaipio and Somersalo, 2005)
    - Support for the builtin probabilistic modeling tool of UQLAB for flexible prior specification
    - Custom prior specifications
    - Standard and custom error models
    - Numerous solver/sampler strategies
        - Maximum a posteriori (MAP)
        - Metropolis-Hastings MCMC
        - Adaptive MCMC (Haario et al., 2001)
        - Hamiltonian MCMC (Neal, 2011)
        - Affine-transform ensemble MCMC (Goodman and Weare, 2010)

With numerous additional features in the final stages of development, including an overhauled probabilistic input module that supports vine copulas and data-driven inference, a reliability-based design optimization module, high-performance computing support, and much more, UQLAB caters to a broad audience of potential users.

4. NEXT ON THE MENU: ROADMAP TOWARD A GLOBAL APPLIED UQ COMMUNITY

Based on the widespread and rapidly growing adoption of UQLAB in several applied contexts in the past few years (see Section 2), its reception by the intended audience has been overall positive. Indeed, with a steadily increasing number of active users, we feel that a critical mass of "expert UQLAB users" has been reached. In part due to their own backgrounds, and in part thanks to the learning process they were exposed to by UQLAB itself, they could become an invaluable resource to others who are following similar paths.

It is with this mindset that the UQLAB project, which has a much broader scope than merely providing a useful software for interested researchers, enters its next phase. With the goal of bringing together users from over 70 countries, with different backgrounds but a common need of performing/introducing UQ in their field, we intend to start a global applied UQ community, beginning from the current UQLAB userbase.

While until now the development of the software has been mostly centralized, so as to "educate" a strong userbase to proper UQ practices, in the upcoming years we intend to move instead to a more user-centric development model. In such a model, users can propose improvements and bugfixes to the current features, develop and share their own features with other users, and provide expert user support in a much more timely manner than in the current centralized model. They could also get involved in steering the software development process by means of feature requests, polls, etc.

Our starting point is to provide a place for the current users to interact with each other and with the developers. It is the community website, `uqworld.org`, which includes the following communication channels:

- *Communication from the project leadership to the community*: in the form of a blog, which is a communication channel originating from the project leaders, covering more general topics such as good practices in UQ (e.g., what is the best technique to use for which application? How to properly pose a certain class of problems?). This channel is mostly unidirectional and used as a sort of "editorial" content, but allows the users to interact within the comment section. It belongs to the original "dissemination/education" mission aspect of the UQLAB project.
- *Communication between developers and users*: in the form of a forum with multiple topics, where the developers share information about the current release (e.g. the current release notes), upcoming features, as well as handling polls for new features. This is a bidirectional communication channel that allows users to actively and directly participate in the design process by commenting on existing features, as well as voting on or proposing new ones.

- *Communication between the users*: in the form of a forum with multiple topics. This section of the website encourages users to interact with each other. It includes the classical *users helping users* section, where experienced users provide support to new users, but also a file-sharing section, where users share their own content, as well as free-form discussion forums. This is a bidirectional communication channels, where users, developers and leadership could easily interact and exchange opinions and experiences.

While the second point is related to the UQLAB software itself, the first and the third communication channels are expected to evolve into a more general (not UQLAB-related) community hub that gathers expertise in applied UQ from different fields, perspectives, and know-how, something that, as of the time of writing, is sorely missing in the highly fragmented landscape of UQ practitioners.

5. CONCLUSIONS

After achieving a critical mass of users, UQLAB has reached maturity as a well-established UQ software, and the next phase of its development has begun. The first steps toward the ambitious objective of creating a worldwide applied UQ community have been made by leveraging the success of the UQLAB software and its userbase. It is our hope that within the upcoming years a highly accessible online community of experts in applied UQ will be established, so that UQ will become a standard tool for a much wider audience than it is at present.

6. REFERENCES

Adams, B. M., Bauman, L. E., Bohnhoff, W. J., Dalbey, K. R., Ebeida, M. S., Eddy, J. P., Eldred, M. S., Hough, P. D., Hu, K. T., Jakeman, J. D., Stephens, J. A., Swiler, L. P., Vigil, D. M., , and Wildey, T. M. (2014). *Dakota, A Multilevel Parallel Object-Oriented Framework for Design Optimization, Parameter Estimation, Uncertainty Quantification, and Sensitivity Analysis: Version 6.0 User's Manual*. Sandia National Laboratories. Technical Report SAND2014-4633 (Updated November 2015 (Version 6.3)).

Andrianov, G., Burriel, S., Cambier, S., Dutfoy, A., Dutka-Malen, I., de Rocquigny, E., Sudret, B., Benjamin, P., Lebrun, R., Mangeant, F., and Pendola, M. (2007). "Open TURNS, an open source initiative to Treat Uncertainties, Risks'N Statistics in a structured industrial approach." *Proc. ESREL'2007 Safety and Reliability Conference, Stavenger, Norway*.

Blatman, G. and Sudret, B. (2011). "Adaptive sparse polynomial chaos expansion based on Least Angle Regression." *J. Comput. Phys*, 230, 2345–2367.

Borgonovo, E. (2007). "A new uncertainty importance measure." *Reliab. Eng. Sys. Safety*, 92, 771–784.

COMSOL, Inc (2018). *COMSOL Multiphysics reference manual, version 5.4*, <www.comsol.com.>.

Cotter, S. (1979). "A screening design for factorial experiments with interactions." *Biometrika*, 66(2), 317–320.

E. De Rocquigny, N. Devictor, and S. Tarantola, eds. (2008). *Uncertainty in industrial practice – A guide to quantitative uncertainty management*. John Wiley & Sons.

Ditlevsen, O. and Madsen, H. (1996). *Structural reliability methods*. J. Wiley and Sons, Chichester.

Echard, B., Gayton, N., and Lemaire, M. (2011). "AK-MCS: an active learning reliability method combining Kriging and Monte Carlo simulation." *Structural Safety*, 33(2), 145–154.

Goodman, J. and Weare, J. (2010). "Ensemble samplers with affine invariance." *Comm. App. Math. Com. Sc.*, 5(1), 65–80.

Haario, H., Saksman, E., and Tamminen, J. (2001). "An adaptive Metropolis algorithm." *Bernoulli*, 7(2), 223–242.

Kaipio, J. and Somersalo, E. (2005). *Statistical and Computational Inverse Problems*. Number 160 in Applied Mathematical Sciences. Springer, New York.

Konakli, K. and Sudret, B. (2016). "Polynomial metamodels with canonical low-rank approximations: Numerical insights and comparison to sparse polynomial chaos expansions." *J. Comput. Phys.*, 321, 1144–1169.

Kucherenko, S., Tarantola, A., and Annoni, P. (2012). "Estimation of global sensitivity indices for models with dependent variables." *Comput. Phys. Comm.*, 183, 937–946.

Le Gratiet, L., Marelli, S., and Sudret, B. (2017). *Metamodel-based sensitivity analysis: polynomial chaos expansions and Gaussian processes*. Springer, Chapter 8, Handbook on Uncertainty Quantification (Ghanem, R. and Higdon, D. and Owhadi, H. (Eds.).

Lebrun, R. and Dutfoy, A. (2009). "An innovating analysis of the Nataf transformation from the copula viewpoint." *Prob. Eng. Mech.*, 24(3), 312–320.

Lemaire, M. (2009). *Structural reliability*. Wiley.

Marelli, S. and Sudret, B. (2014). "UQLab: A framework for uncertainty quantification in Matlab." *Vulnerability, Uncertainty, and Risk (Proc. 2nd Int. Conf. on Vulnerability, Risk Analysis and Management (ICVRAM2014), Liverpool, United Kingdom)*, 2554–2563.

Neal, R. M. (2011). "MCMC using Hamiltonian dynamics." *Handbook of Markov Chain Monte Carlo*, S. Brooks, A. Gelman, G. L. Jones, and X.-L. Meng, eds., Handbooks of Modern Statistical Methods, Chapman & Hall/CRC, Boca Raton, Florida, USA, Chapter 5, 113–162.

Nelsen, R. B. (2006). *An introduction to copulas*, Vol. 139 of *Lecture Notes in Statistics*. Springer-Verlag, New York, 2nd edition.

Patelli, E., Broggi, M., de Angelis, M., and Beer, M. (2014). "OpenCossan: an efficient open tool for dealing with epistemic and aleatory uncertainties." *Vulnerability, Uncertainty, and Risk (Proc. 2nd Int. Conf. on Vulnerability, Risk Analysis and Management (ICVRAM2014), Liverpool, United Kingdom)*.

Rasmussen, C. and Williams, C. (2006). *Gaussian processes for machine learning*. Adaptive computation and machine learning. MIT Press, Cambridge, Massachusetts.

Saltelli, A., Ratto, M., Andres, T., Campolongo, F., Cariboni, J., Gatelli, D., Saisana, M., and Tarantola, S. (2008). *Global Sensitivity Analysis – The Primer*. Wiley.

Santner, T. J., Williams, B. J., and Notz, W. I. (2003). *The Design and Analysis of Computer Experiments*. Springer, New York.

Schöbi, R., Sudret, B., and Marelli, S. (2017). "Rare event estimation using Polynomial-Chaos-Kriging." *ASCE-ASME J. Risk Uncertainty Eng. Syst., Part A: Civ. Eng*, 3(2) D4016002.

Schöbi, R., Sudret, B., and Wiart, J. (2015). "Polynomial-chaos-based Kriging." *Int. J. Uncertainty Quantification*, 5(2), 171–193.

Schölkopf, B. and Smola, A. (2002). *Learning with kernels*. MIT Press.

Smith, M. (2009). *ABAQUS/Standard User's Manual, Version 6.9*. Simulia.

Sobol', I. (2001). "Global sensitivity indices for nonlinear mathematical models and their Monte Carlo estimates." *Math. Comput. Simul.*, 55(1-3), 271–280.

Sobol', I. M. (1993). "Sensitivity estimates for nonlinear mathematical models." *Math. Modeling & Comp. Exp.*, 1, 407–414.

Sudret, B. (2007). *Uncertainty propagation and sensitivity analysis in mechanical models – Contributions to structural reliability and stochastic spectral methods*. Université Blaise Pascal, Clermont-Ferrand, France. Habilitation à diriger des recherches, 173 pages.

Sudret, B. (2008). "Global sensitivity analysis using polynomial chaos expansions." *Reliab. Eng. Sys. Safety*, 93, 964–979.

Sudret, B. and Caniou, Y. (2013). "Analysis of covariance (ANCOVA) using polynomial chaos expansions." *Proc. 11th Int. Conf. Struct. Safety and Reliability (ICOSSAR'2013), New York, USA*, G. Deodatis, ed.

Tarantola, A. (1987). *Inverse problem theory – Methods for data fitting and model parameter estimation*. Elsevier.

Vapnik, V. (1995). *The Nature of Statistical Learning Theory*. Springer-Verlag, New York.

Xiu, D. and Karniadakis, G. E. (2002). "The Wiener-Askey polynomial chaos for stochastic differential equations." *SIAM J. Sci. Comput.*, 24(2), 619–644.